What is Claimed is:

[c1] A method of determining timing windows in a static timing analysis of an integrated circuit design, said method comprising: determining, for at least one node in said integrated circuit design, an initial set of sub-windows; and merging said sub-windows of said initial set into a final set of sub-windows.

[c2] The method in claim 1, wherein said merging comprises merging overlapping sub- windows

[c3] The method in claim 1, further comprising determining for said at least one node a maximum number of sub-windows.

[c4] The method of claim 3, wherein said merging comprises: finding a set of largest intervals between said sub-windows of said initial set, the number of intervals in said set being less than or equal to one less than said maximum number of sub-windows; and combining sub-windows separated by intervals not in said set of largest intervals.

[c5] The method in claim 3, further comprising performing an initial static timing analysis to produce a maximum window for said at least one node.

[c6] The method in claim 5, wherein said maximum number of sub-windows is based on the size of said maximum window.

[c7] The method of claim 5, wherein said sub-windows are based on an equal division of said maximum window.

[c8] The method of claim 5, wherein said maximum window extends from a beginning of an earliest input signal to said node to an ending of latest input to said node.

[c9] The method in claim 1, wherein each sub-window of said initial set of sub-windows extends an earliest beginning of a first input to said node to a latest ending of said first input.

[c10] The method in claim 1, wherein said timing windows are signal switching windows and computation of said windows is performed in a direction in which signals travel through said integrated circuit.

[c11] The method in claim 1, wherein said timing windows are victim windows and the computation of said windows is performed in a direction opposite to that in which signals travel through said integrated circuit.

[c12] A method of determining timing windows in a static timing analysis of an integrated circuit design, said method comprising: determining, for at least one node in said integrated circuit design, an initial set of sub-windows; merging said sub-windows of said initial set into a final set of sub-windows by finding a set of largest intervals between said sub-windows of said initial set, the number of intervals in said set being less than or equal to one less than said maximum number of sub-windows; and combining sub-windows separated by intervals not in said set of largest intervals.

[c13] The method in claim 12, wherein said merging comprises merging overlapping sub- windows.

[c14] The method in claim 12, further comprising determining for said at least one node a maximum number of sub-windows.

[c15] The method in claim 14, further comprising performing an initial static timing analysis to produce a maximum window for said at least one node.

[c16] The method in claim 15, wherein said maximum number of sub-windows is based on the size of said maximum window.

[c17] The method of claim 15, wherein said sub-windows are based on an equal division of said maximum window.

[c18] The method of claim 15, wherein said maximum window extends from a beginning of an earliest input signal to said node to an ending of latest input to said node.

[c19] The method in claim 12, wherein each sub-window of said initial set of sub-windows extends an earliest beginning of a first input to said node to a latest ending of said first input.

[c20] The method in claim 12, wherein said timing windows are signal switching windows and computation of said windows is performed in a direction in which signals travel through said integrated circuit.

[c21] The method in claim 12, wherein said timing windows are victim windows and the computation of said windows is performed in a direction opposite to that in which signals travel through said integrated circuit.

[c22] A method of determining whether an adjacent aggressor net may switch during a victim window of a victim net in a static timing analysis of an integrated circuit design, said method comprising: a) checking for overlap between said victim window and a switching window of said aggressor net; b) translating said victim net to a context of each input of said aggressor net, if said checking step identified an overlap; c) repeating said checking for each said input and translated victim window; and d) determining that said aggressor net may switch during said victim window only if said checking step found an overlap and at least one of said repeating steps determined that said input could switch during said translated victim window.

[c23] The method of claim 22, wherein said repeating step is repeated only for a limited number of levels.

[c24] The method of claim 22, wherein each said input comprises an aggressor sub-window, each relating to one input.

[c25] The method of claim 24, wherein said repeating process checks for overlap between each said aggressor sub-window and a related translated victim window.

[c26] The method in claim 22, further comprising, if said checking step identified an overlap: determining an initial set of aggressor sub-windows; and merging said aggressor sub-windows of said initial set into a final set of aggressor sub-windows, wherein said repeating process checks for overlap between said translated victim window and each of said final set of aggressor sub-windows.

[c27] The method in claim 22, wherein said method is performed in a direction in which signals travel through said integrated circuit.

[c28] The method in claim 22, wherein said method is performed in a direction opposite that in which signals travel through said integrated circuit.

[c29] A method of determining whether an adjacent aggressor net may switch during a victim window of a victim net in a static timing analysis of an integrated circuit design, said method comprising: a) checking for overlap between said victim window and a switching window of said aggressor net; b) translating said victim net to a context of each input of said aggressor net, if said checking step identified an overlap; c) determining an initial set of aggressor sub-windows, if said checking step identified an overlap, and merging said aggressor sub-windows of said initial set into a final set of aggressor sub-windows, d) repeating said checking for each said aggressor sub-windows and translated victim window; and e) determining that said aggressor net may switch during said victim window only if said checking step found an overlap and at least one of said repeating steps determined that said input could switch during said translated victim window.

[c30] The method of claim 29, wherein said repeating step is repeated only for a limited number of levels.

[c31] The method of claim 29, wherein each said input comprises an aggressor sub-window, each relating to one input.

[c32] The method of claim 31, wherein said repeating process checks for overlap between each said aggressor sub-window and a related translated victim window.

[c33] The method in claim 29, wherein said method is performed in a direction in which signals travel through said integrated circuit.

[c34] The method in claim 29, wherein said method is performed in a direction opposite that in which signals travel through said integrated circuit.

[c35] A program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform a method of determining timing windows in a static timing analysis of an integrated circuit design, said method comprising: determining, for at least one node in said integrated circuit design, an initial set of sub-windows; and merging said sub-windows of said initial set into a final set of sub-windows.

[c36] The program storage device in claim 35, wherein said merging comprises merging overlapping sub- windows.

[c37] The program storage device in claim 35, wherein said method further comprises determining for said at least one node a maximum number of sub-windows.

[c38] The program storage device of claim 37, wherein said merging comprises: finding a set of largest intervals between said sub-windows of said initial set, the number of intervals in said set being less than or equal to one less than said maximum number of sub-windows; and combining sub-windows separated by intervals not in said set of largest intervals.

[c39] The program storage device in claim 37, wherein said method further comprises performing an initial static timing analysis to produce a maximum window for said at least one node.

[c40] The program storage device in claim 39, wherein said maximum number of sub-windows is based on the size of said maximum window.

[c41] The program storage device of claim 39, wherein said sub-windows are based on an equal division of said maximum window.

[c42] The program storage device of claim 39, wherein said maximum window extends from a beginning of an earliest input signal to said node to an ending of latest input to said node.

[c43] The program storage device in claim 35, wherein each sub-window of said initial set of sub-windows extends an earliest beginning of a first input to said node to a latest ending of said first input.

[c44] The program storage device in claim 35, wherein said timing windows are signal switching windows and computation of said windows is performed in a direction in which signals travel through said integrated circuit.

[c45] The program storage device in claim 35, wherein said timing windows are victim windows and the computation of said windows is performed in a direction opposite to that in which signals travel through said integrated circuit.